Booleans and Comparisons

Python for Absolute Beginners · Lesson 8

What you'll learn

- What a boolean is: True and False
- How comparisons ask yes-or-no questions
- All six comparison operators
- How to store a comparison's answer in a variable
- Why = and == are different, and why mixing them up is the classic beginner bug

True or False: the boolean

- A boolean is a value that can only be one of two things: True or False — no maybe, no in-between
- Named after the mathematician George Boole
- Note the capital letters: Python writes True and False with a capital first letter
- Lowercase true gives a NameError — Python only recognizes the capitalized forms



Booleans in code

```
print(True)
print(False)

Output:
True
False
```

On their own, True and False are not very interesting — they become powerful when a comparison produces one.

Comparisons ask yes-or-no questions

- A comparison compares two values and answers True or False
- == equal to · != not equal to · > greater than · < less than
- >= greater than or equal to · <= less than or equal to
- Comparisons work on strings too, in dictionary (alphabetical) order



Comparisons in code

```
print(10 == 10)
print(10 != 9)
print(7 >= 7)
print(7 <= 6)

Output:
True
False
True
True
```

Each line asks a yes-or-no question, and Python answers True or False.

Comparing strings in code

```
print("apple" < "banana")
print("Sam" == "Sam")

Output:
True
True
```

“apple” comes before “banana” alphabetically, so the first answer is True.

Storing the answer

- A comparison's answer is just a value, so it can live in a variable like anything else
- Storing the answer separately keeps the decision readable — the variable's name says what the question meant
- In Lesson 9, you will use variables like this to make decisions

Storing the answer in code

```
                              age = 12
old_enough = age >= 10
print(old_enough)

Output:
True
```

old_enough now holds True — the answer to “is age 10 or older?”

The classic bug: = versus ==

- = stores a value: age = 12 puts 12 into age
- == asks whether two values are equal: age == 12 answers True or False
- Using one = where you meant two does not ask a question — it stores a value, usually by accident
- If a comparison ever seems to “always be true,” check whether you wrote = instead of == first

Hands-on exercise

- In the shell, work out each of these before running them, then check: 5 > 3, 5 < 3, 5 == 5, 5 != 5, 4 >= 4, 4 <= 3
- Store your age in a variable, then store the answer to “am I 10 or older?” in a second variable and print it
- Compare two names with == — once with the same name twice, once with different names — and explain both results

Key takeaways

- A boolean is True or False, always capitalized
- Comparisons (==, !=, >, <, >=, <=) answer yes-or-no questions
- A comparison's answer can be stored in a variable for later use
- = stores; == asks. Mixing them up is the classic beginner bug

Take the quiz

Take the Lesson 8 quiz — 5 quick questions to check what you learned.

Next lesson

If Statements: Making Decisions